IPAS-BUS 21.04.2009

HIT: 1 OF 2, Selected: 0 OF 0

© Thomson Scientific Ltd. DWPI

Accession Number
2000-105718

Title Derwent
Redundancy managing method for computer system

Abstract Derwent
Novelty: Redundancy management systems (RMS) provided in several computing nodes respectively, are connected via a communication link. The fault tolerant executive (FTE) model (13) is implemented in each RMS to synchronize the nodes with clock signal, for managing faults and various system functions.
Description: An INDEPENDENT CLAIM is also included for redundancy managing apparatus.
Use: For managing fault tolerant computing used in computing environment for e.g. aerospace, critical control system, telecommunication, computer networks.
Advantage: Additional flexibility is provided in the distributed computing environment by not interweaving with the application. System fault tolerance is achieved by detecting and masking erroneous data through voting, and system integrity is ensured by dynamically reconfigurable architecture which excludes faulty nodes from the system and readmits corrected nodes back into the system.
Description of Drawing: The figure shows block diagram of redundancy managing system.FTE model (13)

Assignee Derwent + PACO
ALLIED-SIGNAL INC ALLC-S

Inventor Derwent

BOLDUC L P	ERNST J W
PENG D	RODEN T G
YOUNIS M	ZHOU J X

Patent Family Information

WO1999063440-A1	1999-12-09	AU9946734-A	1999-12-20
US6178522-B1	2001-01-23	EP1082661-A1	2001-03-14
CN1311877-A	2001-09-05	JP2002517819-W	2002-06-18
TW486637-A	2002-05-11	CN1192309-C	2005-03-09

First Publication Date 1999-12-09

Priority Information
US000140174 1998-08-25 US000087733P 1998-06-02

Derwent Class
T01 W01 W06

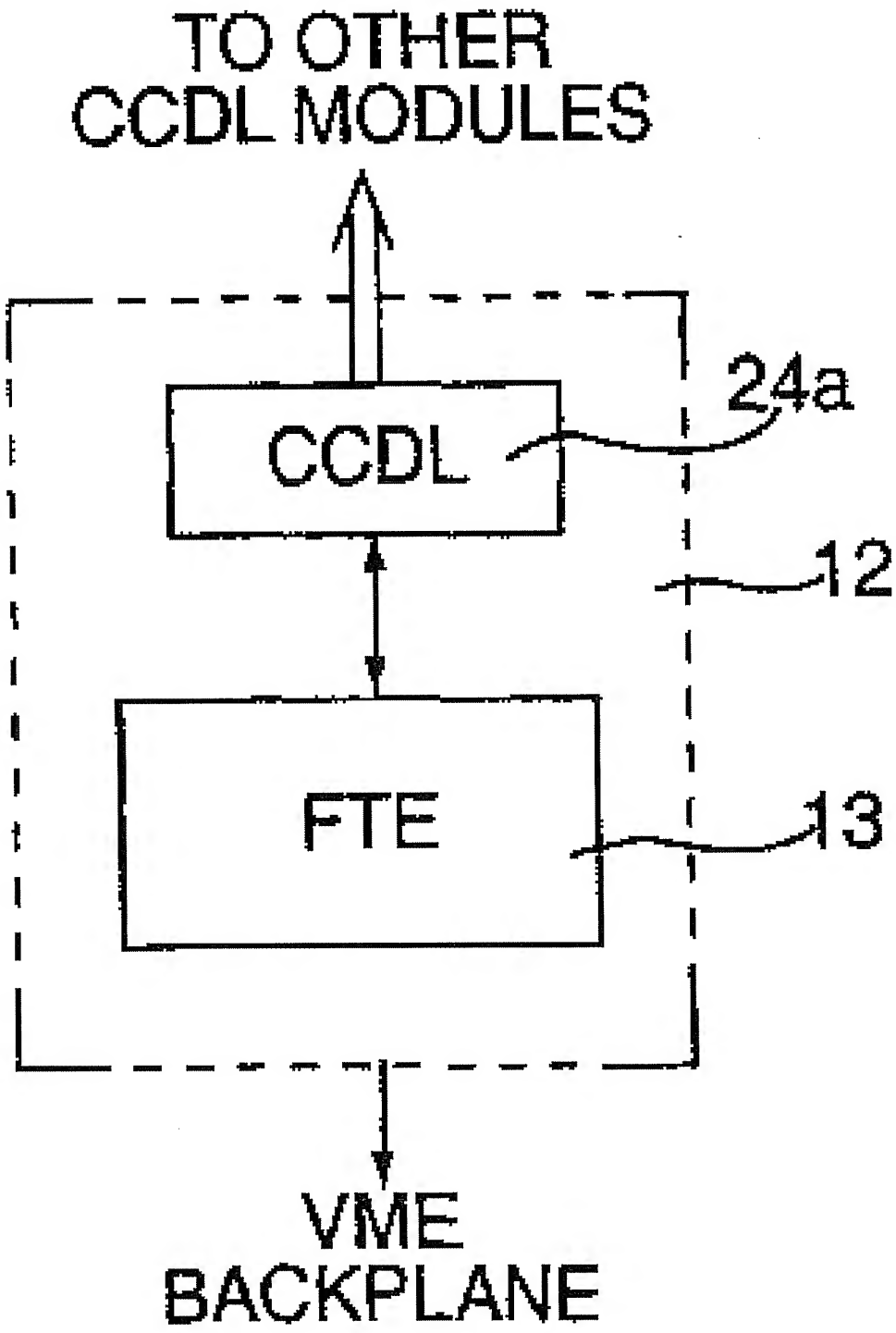
Manual Code

T01-F05B	T01-G05A	T01-H07C5A
W01-A01	W06-B01A	

International Patent Classification (IPC)

IPC Symbol	IPC Rev.	Class Level	IPC Scope
G06F-11/00	2006-01-01	I	C
G06F-11/18	2006-01-01	I	C
G06F-15/16	2006-01-01	I	C
G06F-9/46	2006-01-01	I	C
G06F-11/00	2006-01-01	I	A
G06F-11/18	2006-01-01	I	A
G06F-15/177	2006-01-01	I	A
G06F-9/52	2006-01-01	I	A
G06F-11/18	-		
G06F-17/00	-		

Drawing


TO OTHER
CCDL MODULES
CCDL
24a
12
FTE
13
VME
BACKPLANE

[19]中华人民共和国国家知识产权局

[51]Int. Cl[7]
G06F 11/00

[12]发明专利申请公开说明书

[21]申请号 99809290.8

[43]公开日 2001年9月5日

[11]公开号 CN 1311877A

[22]申请日 1999.6.2 [21]申请号 99809290.8
[30]优先权
[32]1998.6.2 [33]US [31]60/087,733
[32]1998.8.25 [33]US [31]09/140,174
[86]国际申请 PCT/US99/12000 1999.6.2
[87]国际公布 WO99/63440 英 1999.12.9
[85]进入国家阶段日期 2001.2.2
[71]申请人 联合讯号公司
地址 美国新泽西州
[72]发明人 J·X·周 T·G·罗登三世
L·P·波尔杜克 D·-T·彭
J·W·埃恩斯特 M·尤尼斯

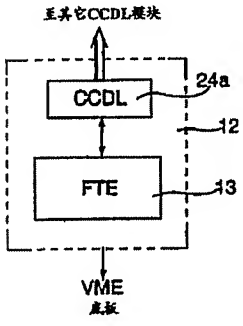
[74]专利代理机构 中国专利代理(香港)有限公司
代理人 吴立明 张志醒

权利要求书3页 说明书13页 附图页数11页

[54]发明名称 管理冗余的基于计算机的系统用于容错计算的方法和设备

[57]摘要

一个独立的冗余管理系统(RMS)(12)以实现极高的系统可靠性，安全性，容错能力和任务成功率，提供了一个管理基于冗余计算机的 系统的有成本效益的解决方案。RMS 包括一个交叉通道数据链结（CCDL）模块(24a)和一个容错执行(FTE)模块(13)。CCDL 模块 提供所有通道的数据通信，同时 FTE 模块执行系统功能，如同步，数 据表决，故障和错误检测，隔离和恢复。系统容错能力通过数据表决 由检测和屏蔽有故障数据来实现的，系统完整性由一个动态重新配置 结构来保证的，该结构它能从系统中排除有故障节点并再许可健康节返回系统中。



ISSN1008-4274

01·02·02

权 利 要 求 书

1. 一种管理基于冗余计算机的有多个硬件计算节点（通道）的系统的方法，包括步骤：

提供在每个计算节点中的冗余管理系统（RMS）；

建立在每个 RMS 之间的通信链结；和

在每个 RMS 中实现容错执行（FTE）模块以管理错误和多个系统功能。

2. 权利要求 1 中的方法，还包括同步系统中的每个计算节点的步骤，所述同步步骤由 FTE 模块执行并包括步骤：

在每个 RMS 提供一个时钟；

在每个 RMS 与所有其它节点交换本地时间；和

根据一个表决的系统时钟调整各自的每个 RMS 的本地时钟。

3. 权利要求 1 中的方法，还包括检测在一个节点中生成的数据中的故障/错误的步骤和防止繁殖在一个节点中生成的数据中的被检测出的故障/错误的步骤，所述检测和防止的步骤还包括步骤：

对每个节点生成的数据表决以决定一个节点产生的数据是否不同于大多数；和

当由一个特殊节点生成的数据不同于表决出的大多数时，使用表决出的数据作为一个输出以屏蔽故障。

4. 权利要求 1 中的方法，其中所述在每个计算节点提供 RMS 的步骤独立于应用开发而执行。

5. 权利要求 1 中的方法，其中所述建立步骤是与在每个计算节点的 RMS 之间的一个交叉通道数据链结（CCDL）合作执行的。

6. 权利要求 1 中的方法，还包括步骤：

定义每个计算节点（通道）为一个故障封锁区；

检测在一个计算节点中生成的数据中的故障/错误；和

隔离在故障封锁区内的一个被检测出的故障以防止繁殖到另一个计算节点。

7. 权利要求 6 中的方法，其中所述检测步骤还包括对每个节点生成的数据进行表决的步骤以决定由某个节点生成的数据是否不同于被表决出的大多数。

8. 权利要求 7 中的方法，其中所述隔离步骤还包括当由一个特殊节点生成的数据不同于表决出的大多数时，使用被表决出的数据作为一个输出来屏蔽故障。

9. 权利要求 3 中的方法，还包括步骤：

确认一个有故障节点响应数据表决的结果；

由全局惩罚系统惩罚被确认的有故障节点；和

当这个有故障节点的惩罚超过用户指定的故障容忍范围时，从节点的一个运行集合中排除这个被确认的有故障节点。

10. 权利要求 9 中的方法，还包括步骤：

监视在被排除节点上的数据以决定这个被排除的节点是否有资格再许可进入一个运行集合；和

当监视表明该节点的表现在预定限度的可接收范围时，再许可该被排除节点进入运行集合。

11. 权利要求 10 中的方法，其中预定限度由系统操作员定义。

12. 一种在有多个计算节点（通道）的计算环境中用于容错计算的方法，包括步骤：

在每计算节点独立于应用实现一个冗余管理系统（RMS）；

在每个 RMS 之间通信；和

维护节点的运行集合用于增加计算环境的容错性。

13. 权利要求 12 中的方法，其中所述通信步骤在一个交叉通道数据链上（CCDL）执行。

14. 权利要求 13 中的方法，其中所述通信步骤还包括：

CCDL 与相应的 RMS 的节点接口；

提供多个在 CCDL 中的接收器，以从多个节点中的每一个分别接收数据；

提供至少一个在 CCDL 中的传送器，以处理和传送接收到的数据至 RMS 中的容错执行驻留。

提供至少一个接收器存储器和至少一个传送器存储器来按需要接收和保存各自的数据。

15. 权利要求 12 中 的方法，其中所述维持节点的运行集合是执行在 RMS 中的一个容错执行住户中，并且还包括步骤：

从连接到计算环境中的每一个节点接收数据；

决定从任何一个节点接收到的数据是否包含故障；

排除生成的数据对于其它接收到的数据有故障的一个节点；和

重新配置运行集合不包括有故障节点。

16. 权利要求 15 中的方法，其中所述决定步骤包括步骤：

为有故障数据设置一个容忍范围；

对从每个决定接收到的数据表决；

确认一个故障数据超过设置的容忍范围的节点。

17. 权利要求 15 中的方法，还包括步骤：

监视被排除节点上的数据；和

当被监视数据表示在被排除节点上的有故障数据校正时，再许可被排除的节点进入运行集合。

18. 权利要求 16 中的方法，其中所述表决步骤是在数据传送中执行在每个次要帧边界上。

19. 权利要求 15 中的方法，其中所述重新配置步骤是在数据传送中执行在每个主要帧边界上。

20. 一种管理基于冗余计算机的有多个硬件计算节点（通道）的系统的设备包括：

在每个计算节点提供冗余管理系统（RMS）的装置；

建立在每个 RMS 之间的通信链结的装置；和

实现在每个用于管理故障和多个系统功能的容错执行模块的装置。

21. 权利要求 20 中的设备，其中所述建立通信链结的装置还包括一个连接至每个计算节点中的每个冗余管理系统的交叉通道数据链结。

22. 权利要求 20 中的设备，还包括：

检测任何一个计算节点中生成的数据中的故障/错误的装置；和

在故障/错误生成的节点内隔离故障/错误的装置。

23. 权利要求 22 中的方法，其中所述检测装置还包括对每个节点生成的数据进行表决的装置以决定某个节点生成的数据是否不同于被表决出的大多数。

24. 权利要求 23 中的方法，其中所述隔离装置还包括使用表决出的数据来屏蔽由不同于表决出的大多数的某个节点产生的一个故障。

说　明　书

管理冗余的基于计算机的系统用于容错计算的方法和设备

本发明涉及计算环境，更特别地，涉及管理冗余的基于计算机的系统用于容错计算的方法。

容错计算在一个系统存在故障和错误的情况下保证正确的计算结果。冗余使用是容错的主要方法。有许多不同的方式管理在硬件，软件，信息和时间上的冗余。由于各种各样的算法和实现手段，大多数现有系统使用冗余管理的使用权设计，这些设计一般与应用软件和应用硬件混杂在一起。应用与冗余管理的混杂产生一个更为复杂的系统，它显著降低了灵活性。

因而本发明的目的是提供用于管理冗余的基于计算机的系统，它不与应用参杂在一起，并提供在分布计算环境中的附加的灵活性。

根据本发明的一个实施例，通过在一个分布环境中使用多个硬件计算节点（或通道）和在每个单独节点上安装冗余管理系统（RMS）构造这种冗余计算系统。

RMS 是通过经过在每个计算系统中的处理单元应用的算法集，数据结构，运行过程和设计而实现的冗余管理方法学。RMS 在许多需要高度系统可靠性的领域，如航空，关键控制系统，电讯，计算机网络等等中有宽广的应用。

为了实现 RMS，RMS 从应用开发物理地或逻辑地被分开。这减少了将来系统的整体设计的复杂性。因此，系统开发者能独立地设计应用，并依靠 RMS 提供冗余管理功能。RMS 和应用的集成是通过可编程的连接 RMS 与应用处理器的总线接口协议来完成的。

RMS 包括一个交叉通道数据链接（CCDL）模块和一个容错执行（FTE）模块。CCDL 模块为所有通道提供数据通信，同时 FTE 模块提供诸如同步，表决，故障和错误检测，隔离和恢复的系统功能。通过表决来检测和屏蔽错误数据实现系统容错，动态重新配置机制，即能够排除系统中的有故障节点并再次许可健康节点返回系统中，保证系统完整性。

RMS 能用硬件，软件或二者的结合（如混合）来实现，并在一个有冗余计算资源来处理元件故障的分布系统中工作。根据系统可靠性和容错要求，分布系统有二个至八个通道（或节点）。一个通道由一个 RMS 和一个应用处理器组成。通道通过 RMS 的 CCDL 模块互相连接形成一个冗余系统。因为在一个通道内的单个应用不充分了解其它通道的活动，所以 RMS 提供系统同步，保持数据一致性，并形成对出现在系统的不同地点的故障和错误的系统范围内的审查。

参考下面详细描述同时结合附图，对本发明的更完整的理解和其中的许多服务优点将是显然的，同样也变得更好理解，其中相似的标号表示同样或相似部件，其中：

图 1 是依照本发明的实施例的冗余管理系统的方框图；

图 2 是依照本发明的示范性实施例的基于三个通道 RMS 的容错系统的方框图；

图 3 是依照本发明的实施例的冗余管理系统的状态转换图；

图 4 是依照本发明的实施例的冗余管理系统，应用交互和表决过程的方框图；

图 5 是依照本发明的实施例的容错执行关系的示意图；

图 6 是依照本发明的实施例容错机执行的表决和惩罚分配过程的方框图；

图 7 是依照本发明的实施例的冗余管理系统关系的示意图；

图 8 是依照本发明的实施例的交叉通道数据链接消息结构的图表；

图 9 是依照本发明的实施例的交叉通道数据链接顶层结构的方框图；

图 10 是依照本发明的实施例的交叉通道数据链接发送器的方框图；

图 11 是依照本发明的实施例的交叉通道数据链接接收器的方框图；

依照本发明的实施例，冗余管理系统（RMS）提供下列冗余管理功能：1〕交叉通道数据通信；2〕基于帧的系统同步；3〕数据表决；4〕故障和错误检测，隔离和恢复；5〕很好的降级和自我复原。

交叉数据通信功能由 CCDL 模块提供。CCDL 模块有一个发送器和

至多 8 个并行接收器。它从它的本地通道取得数据并向所有通道包括自己广播数据。通信数据打包成某种消息格式，使用奇偶位来检测传送错误。为了保护通道之间的电绝缘，所有的 CCDL 接收器使用电光转换。因此，没有单个接收器失败能从其它的通道的接收器过度耗尽电流，导致系统范围的普遍模式失败。

RMS 是一个以帧为基础的同步系统。每个 RMS 有它自己的时钟，通过同所有通道交换它的本地时间并根据表决的时钟并调整它的本地时钟来完成系统同步。使用分布的协定算法，从任何类型的故障包括拜占庭(byzantine)故障引起的失败，建立一个全局时钟。

RMS 使用数据表决作为它的故障检测，隔离和恢复的主要机制。如果一个通道产生的数不同于被表决的大多数的数据，被表决的数据将被使用作为输出来屏蔽故障。有故障的通道将被确认并被全局惩罚系统惩罚。数据表决包括应用数据和系统状态数据二者。RMS 支持不同种类的计算系统，其中由于各种各样的硬件和软件，无故障的通道并不保证生成完全一样的数据（包括数据图象）。如果数据偏离发生在表决过程中，用户指定的容忍度决定错误的行为。

通过从定义操作集合的同步的无故障的一组通道中排除一个失败通道，RMS 支持体面的降级。设计惩罚系统来惩罚由任何有故障的通道提交的错误行为。当一个有故障的通道超出它的惩罚极限时，其它的无故障通道重新配置他们自己为一个新的操作集合，它排除了刚刚确认的有故障通道。这个被排除的通道不容许参与数据表决，它的数据只用来监视目的。RMS 也有能力通过动态重新配置再次许可健康通道回到操作集合中。这种自我康复功能容许 RMS 保护用于长期任务的系统资源。

图 1 示出了依照本发明的实施例的 RMS 系统的顶层方框图。RMS12 包括一个交叉通道数据链接（CCDL）模块 24a，以及一个容错机执行模块 13。FTE13 驻留在 VME 卡或其它单板计算机上，并经过 VME 底板总线或其它适当的数据总线连接至系统中的其它卡。RMS12 经过 CCDL 模块 24a 连接至每个驻留在卡上的其它 RMS 模块。每个 RMS 包括它自己的 CCDL 模块，用于在各自的 RMS 模块中建立一个通信链接。经过 CCDL 的一个通信链接的建立在监视系统中的所有卡的完整性中提供附加的灵活性。通过在每个计算节点上实现 RMS，并彼此连接，系统故障能

被检测，隔离并恢复，它比其它的容错系统更具效率。

图 2 描述了根据本发明的实施例一个示范性的三通道 RMS 基系统结构。在该结构中，RMS 与 3 个车辆任务计算机（VMC）相互连接形成一个冗余的容错的系统。每个 VMC 有一个置于其中的有几个单板机的 VME 底盘。RMS12 安装在 VMC1 的第一个槽中，在 RMS 和其它应用板之间的通信通过 VME 底板总线 14。每个 VMC 从它的外部 1553 总线取得输入。这 3 个主要应用，车辆子系统管理员 16，飞行管理员 18 和任务管理员 20，计算它们的功能，然后保存关键数据于 VME 全局存储器（见图 7）中用于表决。

各个板 VMC1，VMC2 和 VMC3 的每个 RMS12，22 和 32 经过 VME 总线获得数据并通过交叉通道数据链结（CCDL）24 向其它通道广播本地数据。接收数据的 3 个拷贝后，RMS 将表决并将表决的数据回写至 VME 全局存储器以备应用使用。

系统容错

RMS 中的每个通道被定义为一个用于故障检测，隔离和恢复的故障牵制区（FCR）。传统上，FCR 通常有一个被普通的硬件/软件元件限制的边界。FCR 的关键属性是它的防止故障和故障繁殖到另一个区的能力。发生在同一区的多个故障被视为单一故障，因为其它区能通过表决过程检测并校正该故障。一个能容忍的同时故障的数量取决于该系统中无故障通道的数量。对于非拜占庭故障，N 2f + 1，其中 N 是无故障通道的数量，f 是故障数量。如果要求一个系统为拜占庭式安全，N $3f_B + 1$，其中 f_B 是拜占庭故障的数量。

RMS 能容忍有不同时间长度的故障，如瞬间故障，间歇故障和永久故障。瞬间故障有很短的持续时间并随机出现和消失。间歇故障以某种频率间断性地出现和消失。如果不采取矫正行动，永久故障不确定地保持存在。在传统的容错系统设计中，故障元件的严格修剪(pruning)能缩短故障潜伏时间，并因此增强系统的完整性。而且，瞬间故障元件的立即排除可能降低系统资源太快，并危害任务成功。为了平衡这 2 种冲突要求，根据应用需要，RMS 的容错容许用户编程他的惩罚系统。不同的数据和系统错误能赋予不同的惩罚。对某种故障的高惩罚加权将导致当这样的故障发生时的故障通道的迅速排除。对其它故障的低惩罚加权将容许一个有故障通道呆在系统中一个预定时

间，以便它能通过表决修正故障。

根据本发明的 RMS 系统，当惩罚超出用户定义的排除极限时，在 3 节点配置中的故障牵制排除有故障通道。当一个通道的良好行为信用达到再许可极限时，该通道被再许可进入操作集合中。在应用或通道数据中的冲突通过中值选择表决来解决。

在 2 个节点配置中，RMS 不能检测或排除有故障节点。因此，表决不能用于解决冲突。应用必须决定谁有故障并采取相应的行动。

RMS 实现

如前所述，RMS 有 2 个子系统，容错执行（FTE）和交叉通道数据链结（CCDL）。FTE 由 5 个模块（图 5）组成：1〕同步器 80；2〕表决器 58；3〕容错机（FLT）84；4〕任务通信器（TSC）46；5〕内核（KRL）52。这些模块的功能将按前述方法描述。

同步器（SYN）80（图 5）建立和保持系统的通道同步。它要求在任何时间，每个独立的 RMS 必须在或运行于下面 5 个状态之一：1〕POWER_OFF；2〕START_UP；3〕COLD_START；4〕WARM_START；5〕STEADY_STATE；图 2 示出了一个独立的 RMS 和它的 5 个状态的状态转移图。

POWER_OFF（PF）是当 RMS 是非运行的和因为任何理由相关计算机的电源是关闭时的状态。当 RMS 启动时，RMS 无条件地转移到 START_UP 状态。

START_UP（SU）是在计算机已经开启之后，和当所有的系统参数正在初始化，RMS 计时机制正在初始化以及通道间通信链结（如，CCDL）正在建立时的状态。当启动过程完成时，RMS 无条件转移到 COLD_START 状态。

COLD_START（CS）是这样的状态，在该状态，RMS 不能确认一个存在的操作集合（OPS）并正在试图建立一个 OPS。OPS 是参加在正常的系统运行和表决的一组节点。当在 OPS 中少于 3 个 RMS 时，RMS 从 WARM_START 转移至 COLD_START。

WARM_START（WS）是这样的状态，在该状态，RMS 确认包括至少 3 个 RMS 的 OPS，但本地 RMS 自身不在 OPS 中。

STEADY_STATE（SS）是当 RMS 的节点同步于 OPS 时的状态。一个 STEADY_STATE 节点能在或不在 OPS 中。OPS 中的每个节点正在执行它

的正常运行和表决。不包括在 OPS 中的一个节点排除于表决外但它的数据被 OPS 监视，来决定它的再许可进入的资格。

在冷启动状态，使用交互式集中算法来同步通道时钟到集中的为操作集合（OPS）的时钟组之中。要求所有成员对在 OPS 中的成员资格有一致的看法，并且它们在同一时刻也都切换到 Steady_State（稳定状态）模式。

在 Steady_State（稳定状态）模式，每个通道通过系统状态（SS）消息向所有通道广播它的本地时间。为了保持系统同步，每个通道动态调整它的本地时间为全局时间。由于 RMS 是一个帧同步系统，它有一个定义最大可容许同步 SEW 称为软错误窗口（SEW）的预定时间窗口。每个无故障 RMS 应该在 SEW 限定的时间间隔接收其它 SS 消息。由于 RMS 用于分布环境，使用单个 SEW 窗口在判定参与的通道中的同步错误时有天生的不确定性。见 P. Thambidurai, A. M. Kieckhafer 和 C. J. Walter 在 IEEE 第 19 次容错计算国际研讨会发表的“在 MAFT 中的时钟同步”一文，整个内容在这里作为参考。为了解决不确定性，使用另一个已知为硬错误窗口（HEW）的时间窗口。例如，如果通道“A”在“A”的 HEW 之外接收通道“B”的时钟，通道“A”报告通道“B”的同步错误。然而，如果通道“B”发现它自己的时钟（接收它自己的 SS 消息之后）在 HEW 中，通道“B”报告通道“A”有一个有关同步的错误的错误报告。互相责备的通道的不确定性需要通过其它通道对“B”时钟的看法来解决。如果通道“A”是正确的，其它通道应该观察到通道“B”的时钟已经至少达到它们的 SEW 之外。由于其它通道的错误报告存在，于是系统能够确认通道“B”为有故障通道。否则，通道“A”为有故障通道，因为它偏离错误报告中的大多数观点。

热启动（WS）是在冷启动和稳定状态之间的中间状态。由于故障和错误，一个通道可以被排除在 OPS 之外。被排除的通道能经过重置并试图与热启动模式的操作集合再同步。一旦该通道检测到它与操作集合的全局时钟同步，它能切换到稳定模式。一旦处于稳定模式，被排除的通道被监视以备将来再许可进入 OPS。

在 VMC 中的时间同步利用 RMS 生成的本地监视器的中断信号，VSM 日程管理器使用帧边界和中间帧信号来安排任务。

贯穿 VMC 的时间同步保证资源一致。CCDL 时间标记 RMS 系统接收

的数据消息超过 8M 字节数据链。FTE 从 VMC 获取 RMS 系统数据并表决这些接收的消息的时间，调整 CCDL 本地时间为表决值。然后 FTE 生成在已同步的帧边界上的中断信号。

系统表决

在 RMS，表决是用于故障检测，隔离和恢复的主要技术。FTE 中的 RMS 表决器（VTR）对系统状态，错误报告和应用数据进行表决。系统状态的表决建立有关系统运行的一致观点，如 OPS 中的成员资格和同步模式。对错误报告的表决对哪个通道有错误行为和对这些错误应该进行什么惩罚形成一致意见。对应用数据的表决提供正确数据输出以供应用使用。数据表决序列在图 4 中示出。

RMS 数据表决是一个由次要(minor)帧边界驱动的循环操作。一个次要帧边界是在系统中最频繁调用的任务的区间。如图 4 所示，4 个通道的系统生成在次要帧中的应用数据 40，并保存该数据于未加工数据共享的存储器 42，它已知为给 RMS 边界用的应用数据表。在次要帧边界 44，RMS 的任务通信器（TSC）模块 46 使用数据 ID 序列表（DST）48 作为指针来从应用数据表中读取数据。DST48 为一个数据表决程序，它决定哪个数据将需要在每个次要帧中被表决，并且它也包含表决必须的其它相关信息。读取数据之后，TSC46 将该数据打包成某种格式并发送该数据至 CCDL24。CCDL 向其它节点广播它的本地数据，同时也接收来自其它节点的数据。当数据转移完成，内核（KRL）52 从 CCDL24 获得数据并保存该数据于数据备份表 56 中，其中数据的 4 个备份现在准备用作表决（如，来自其它 RMS 的 3 个备份和来自该 RMS 的一个备份）。表决器（VTR）58 执行表决和反常检查。中间值选择算法用作整数型和实型数的表决，大多数表决算法用作二进制和离散型数据表决。数据类型和它的相关误差容忍度也由 DST48 提供，DST48 被 VTR 使用来挑选合适的表决算法。被表决的数据 60 保存在已表决数据表 62 中。在适当的时间，TSC 模块 46 从已表决数据表 62 中读取数据并将它回写至应用数据表（或已表决数据共享存储器）66 中作为表决输出。再者，输出数据的地址由 DST48 提供。如果该系统有 2 个剩下的运行通道并且 VTR 检测到存在数据不一致，那么对于每个被表决的数据，可以由 VTR58 在数据冲突表 64 中设置数据冲突标。数据冲突表 64 位于一个共享存储器空间中，这样应用软件能访问该表，以决定被表决

的数据是否有效。

数据表决选项

数据类型	描述	表决算法	估计表决时间
带符号的整型	32 位整数	中间值选择	6.0 秒
浮点	IEEE 单精度浮点	中间值选择	5.3 秒
不带符号的整型	被表决为一个字的 32 位字（可能在表决状态字时有用）	中间值选择	6.0 秒
32 位表决器	32 位已打包的布尔字，被表决为 32 位独立布尔值	大多数表决	12 秒

表 1

表 1 是数据表决选项的一个示范性表，其中指定的数据类型是 ANSI "C" 语言的 IEEE 标准数据类型。

容错机

通过为每个通道定义故障牵制区，FCR（如通道） 只通过与其它 FCR（通道）的信息交换能表明它的错误。见 J .Zhou 的"设计捕获系统可靠性"，由复杂系统工程综合和评估组出版，NSWC，Silver spring，MD，1992 年 7 月，107－119 页，这里引作参考。通过表决和其它错误检测机制，容错机（FLT）84（图 5）将错误总结为 15 种类型，如表 2 所示。利用 16 位错误矢量来记日志和报告检测出的错误。该错误矢量被打包在一个错误信息中，并向其它通道广播，用于在每一个次要帧的一致化和恢复动作。

错误标识	错误描述	由谁检测	惩罚加权
E1	（保留）		
E2	消息被接收，带有一个非有效的消息类型，节点标识或数据标识	CCDL	1 或 TBD
E3	水平或垂直奇偶错误，不正确的消息长度或消息限制超出	CCDL	1 或 TBD
E4	太多错误报告或系统状态消息被	CCDL	2 或 TBD

	接收		
E5	一个非 SS 消息在硬错误窗口内被接收	KRL	4 或 TBD
E6	从一个节点接收到不只一个同样的数据	KRL	2 或 TBD
E7	丢失 SS 消息，或 PRESYNC/SYNC 消息不是以正确次序达到	SYN	2 或 TBD
E8	一个 SS 消息没有在硬错误窗口（HEW）内达到	SYN	4 或 TBD
E9	一个 SS 消息没有在软错误窗口（SEW）内达到	SYN	2 或 TBD
E10	接收到一个 SS 消息，带有不同于本地节点的一个小型和/或主要帧数目	SYN	4 或 TBD
E11	节点的 CSS 和/或 NSS 不与被表决的 CSS 和/或 NSS 一致	VTR	4 或 TBD
E12	从该次要帧中的节点没有接收到一个错误消息	VTR	2 或 TBD
E13	丢失数据消息	VTR	2 或 TBD
E14	由一个节点生成的数据值与被表决的值不一致	VTR	2 或 TBD
E15	包括在来自一个节点的错误消息中的消息与被表决值不一致	VTR	3 或 TBD
E16	一个主要帧中的一个节点累计的错误数量已超过当前限制	FLT	4 或 TBD

表 2（错误矢量表）

图例:

CSS: 当前系统状态表示在当前次要帧中的 OPS 中的节点

NSS: 下一个系统状态表示在下一个次要帧中的 OPS 中的节点

OPS: 操作集合，它定义为处于稳定状态模式中的无故障系统节点的集合

TBD: 将被决定

CCDL: 交叉通道数据链接

KRL: 内核

SYN: 同步器

VTR: 表决器

FLT: 容错机

参照图 6，FLT84 评定为错误源的一个通道的惩罚 104。在每一个次要帧，所有已检测（已报告）的错误用惩罚加权表 102 来施加惩罚，惩罚总和保存在递增的惩罚计数（IPC）中。本地 IPC 被评定（104）并经过 CCDL 向其它节点广播（106）。FLT 模块对 IPC（108）表决，表决结果保存在基惩罚计数（BPC）110 中。IPC 捕获特别的次要帧的错误，BPC 捕获整个任务时间的累积错误，在计算/保存 BPC（110）之后，IPC 矢量被清空（112），BPC 经过 CCDL 向其它节点广播（114）。为了确保系统重新配置的所有无故障通道之间的动作一致，每一个次要帧也表决 BPC（116）并且 FLT 使用已表决的 BPC 来决定是否需要一个惩罚赋予和表决。一旦对 BPC 的表决（116）完成，FLT 决定是否已经达到一个主要帧边界(118)。如果是的话，该重新配置被决定(120)。如果没有达到主要帧边界，处理返回至错误报告 110，并从头继续。

系统重新配置包括有故障通道的排除和健康通道再许可进入。如果有故障通道的基惩罚计数（BPC）超出一个预定极限，RMS 开始系统重新配置。在重新配置过程中，系统重组操作集合，排除有故障通道。一旦一个通道失去在操作集合中的成员资格，它的时间和系统状态将不再在边界过程中使用。被排除的通道需要进行一个重置处理。如果重置处理成功，该通道能试图再与操作集合同步，如果同步成功的话，它能切换到稳定状态模式。一个被排除的通道能在稳定状态模式中运行，但仍然在操作集合之外。现在该通道从操作集合中的通道接收所有的系统消息和应用数据。

操作集合中的所有成员也接收来自被排除的通道的消息并监视它的行为。根据通道的行为，被排除的通道的 BPC 可以增加和减少。如果被排除的通道保持无故障运行，它的 BPC 将逐渐减少至一个预定的极限，并在下一个主要帧边界，系统进行另一次重新配置以重新许可该通道进入。

RMS 和应用接口

当前 RMS 实现使用 VME 总线和共享存储器作为 RMS 和应用接口。然而，只有一种可能的实现方法，也能利用其它的通信协议来实现该接口。TSC 模块 46（图 4）的主要功能是从指定的通信介质中取得数据并将数据打包成某种格式供 RMS 使用。当一个表决循环完成时，TSC 取得该已表决数据并将该数据返送回应用。

RMS 内核

图 5 示出了依照本发明的实施例的容错执行（FTE）的关系的示意图。如图所示，内核 52 提供对 RMS 的所有监督运行。内核 52 管理 RMS 的启动，调用适当的功能来初始化目标处理器以及所有初始数据的调用。在启动过程中，通过调用系统配置数据和适当的运行参数，内核配置 CCDL 模块。通过监视其它 RMS 模块的状态和在正确时间采取适当动作，内核管理 RMS 操作节点之间的转换（如冷启动，热启动，稳定状态）。内核使用一个确定性的行程算法，使得所有的“动作”由一个自含的时间基地控制。在该时间基地循环的一个给定“记号”处，该记号的预定动作经常被执行。内核 52 基于时间记号调整 FTE 功能。RMS 动作，如错误检测，隔离和恢复，在 RMS 次要帧中的适当时间由内核安排。如果 RMS 通道有故障，内核有责任在适当时间重启该通道。在 RMS 子系统之间和在 RMS 和应用计算机之间的所有数据的转移由内核管理和安排。内核指导其它模块准备各种各样的 RMS 消息并将这些消息调入 CCDL，供在内核需要时传送。只要 CCDL 接收到消息，内核析取那些消息并将它们分派到正确模块来处理。内核循环运行，连续执行每个安排好的动作并监视 RMS 状态。

容错执行（FTE）为 4 个或更多节点提供拜占庭式故障恢复。在来源一致的条件下，给 3 个节点提供拜占庭式安全。FTE 表决应用数据，删除/恢复 FTE 的应用，并同步应用和 FTE 至小于 100 秒。

在一个示范性实施例中，FTE 用大约 4.08 毫秒（利用率 40％）来表决 150 个字并执行操作系统功能。FTE 存储器为 0.4M 个字节的闪存（利用率 5％）和 0.6M 字节的 SRAM（利用率 5％）。提供这些值作为示范用途。本领域的技术人员将理解改变这些值不偏离本发明的范围。

RMS 运行环境

图 7 示出了运行环境中的 RMS 和 VMC 之间的 RMS 关系或交换结构。

在 VMC 内转移的信息包括在 RMS 帧边界交付的 RMS 系统数据，并包括信息，例如次要帧成员，表决过的当前/下一个系统状态表示谁正在运行在运行集之中和之外，系统冲突标用于 2 个节点配置。2 个节点配置中使用数据冲突表来表示一个在同等数据元基础上的不可解决的数据冲突。已表决的输出是为从一个操作集合成员中表决而提交的每个数据元的表决值。RMS 系统数据，数据冲突表和表决过的输出由 RMS 转移至全局共享存储器，它是在与 RMS 正在运行其中的本地 VMC 的通信中。

未经加工的输出是提交给 RMS 的数据，用于在稳定状态模式中被所有节点表决。应用错误计数是该系统的一个可选功能，并被转移至 RMS，用来使一个应用能在决定操作集合时影响由 RMS 评估的错误惩罚。

帧边界信息包括一个发出 RMS 帧开始的信号的中断。这种信号帧同步 FM，VSM 和 MM。中间帧信息是另一种中断，它从帧开始提供一个 5 毫秒的信号。应用数据准备就绪信息包括一个 RMS 产生的中断，向应用发出信号，即已表决的数据正在等待并能被存取和处理。系统重置是一个在重置时应用能够使用的可选控制。

交叉通道链（CCDL）

CCDL 模块提供在通道间的数据通讯。该数据捆绑成消息，该消息结构在图 8 中示出。如图所示，该消息结构包括一个头部，和根据被发送和接收的消息类型的不同的消息类型。消息类型 0 是一个数据消息的结构；类型 1 是一个系统状态消息的结构；类型 2 是一个冷启动消息的结构；类型 4 是一个错误报告和惩罚计数消息的结构。

每个 CCDL 有一个发送器和至多 8 个接收器。CCDL 顶层结构，发送器和接收器图表在图 9－11 中描述。图 9 示出了一个顶层 CCDL 结构，它有一个发送器 70，4 个接收器 72a－72d，和 2 个使用 DY4 MaaxPac 中间层协议的接口 74a 和 74b。一个接口 74b 有助于在基 VME 和 CCDL 存储器之间的数据交换，其它接口 74a 处理控制冲突和错误报告。当数据需要被传送时，CCDL 接口 74b 从基卡中取得数据并将它保存到 8 位发送器存储器 76 中。当数据被接收，4 个接收器 72a－d 处理并一个节点一个地分别保存接收到的数据于 4 个接收器存储器 78a－d 中。然后 FTE 在 CCDL 的控制下获取数据。因为 CCDL 是在通道之间建立物

理连接的唯一模块，为了保证系统的故障牵制区，它必须加强电绝缘。当前的 CCDL 使用电光转换来将电信号转换为光信号。每个接收器 72a－72d 有对应的提供必要的绝缘功能的光隔离 73a－73d。这使得每个通道有它自己的电源供应，并且它们都彼此电绝缘。

图 10 示出了一个已知本发明的实施例的发送器 70 结构的更详细的视图。当由 FTE 发出一个“执行”命令，发送器控制逻辑 80 从它的 8 字节存储器 76 中读取数据，将该数据形成一个 32 位格式，并将一个水平字加到该数据的尾部。转换寄存器电路 82 将该数据转换成一个串行字节串，垂直奇偶位插入到该串中用于传送。

图 11 解释串行字节串是如何从一个发送模式接收并保存在相应的存储器中的。位中心逻辑 90 使用 6 个系统时钟（如，48MHZ）循环来可靠地记录在一个数据位中。当数据串的第一个位被接收，时间标记逻辑 92 记录该时间用于同步目的。转换器电路 94 剥离垂直奇偶位并转换串行数据为 8 位格式。如果垂直位显示传送错误，将报告一个错误。控制逻辑 96 还根据附着在该数据上的节点数信息从该数据中剥离水平奇偶位并将它保存到接收器存储器中（如 78a）。

为了加强通信可靠性，水平和垂直奇偶位附着在数据消息上。消息格式由 CCDL 检验，只有有效消息被发送至内核作进一步处理。

应该理解，本发明不限定于这里公开的特殊实施例，它希望作为实现本发明的最好模式，但本发明不限于本说明书中描述的特定的实施例，而由附加的权利要求书所限制。

说 明 书 附 图

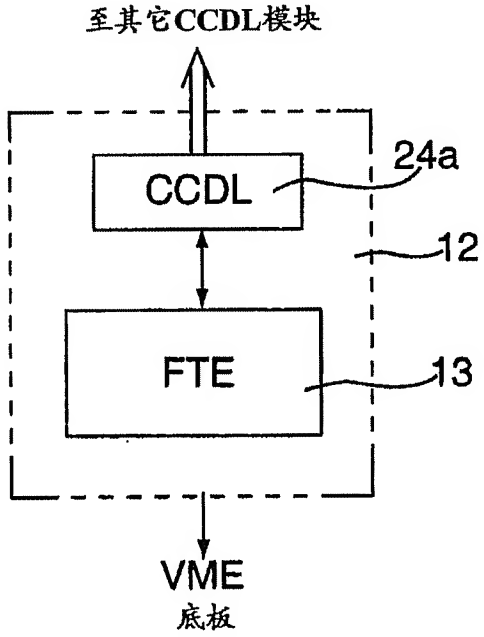


图 1

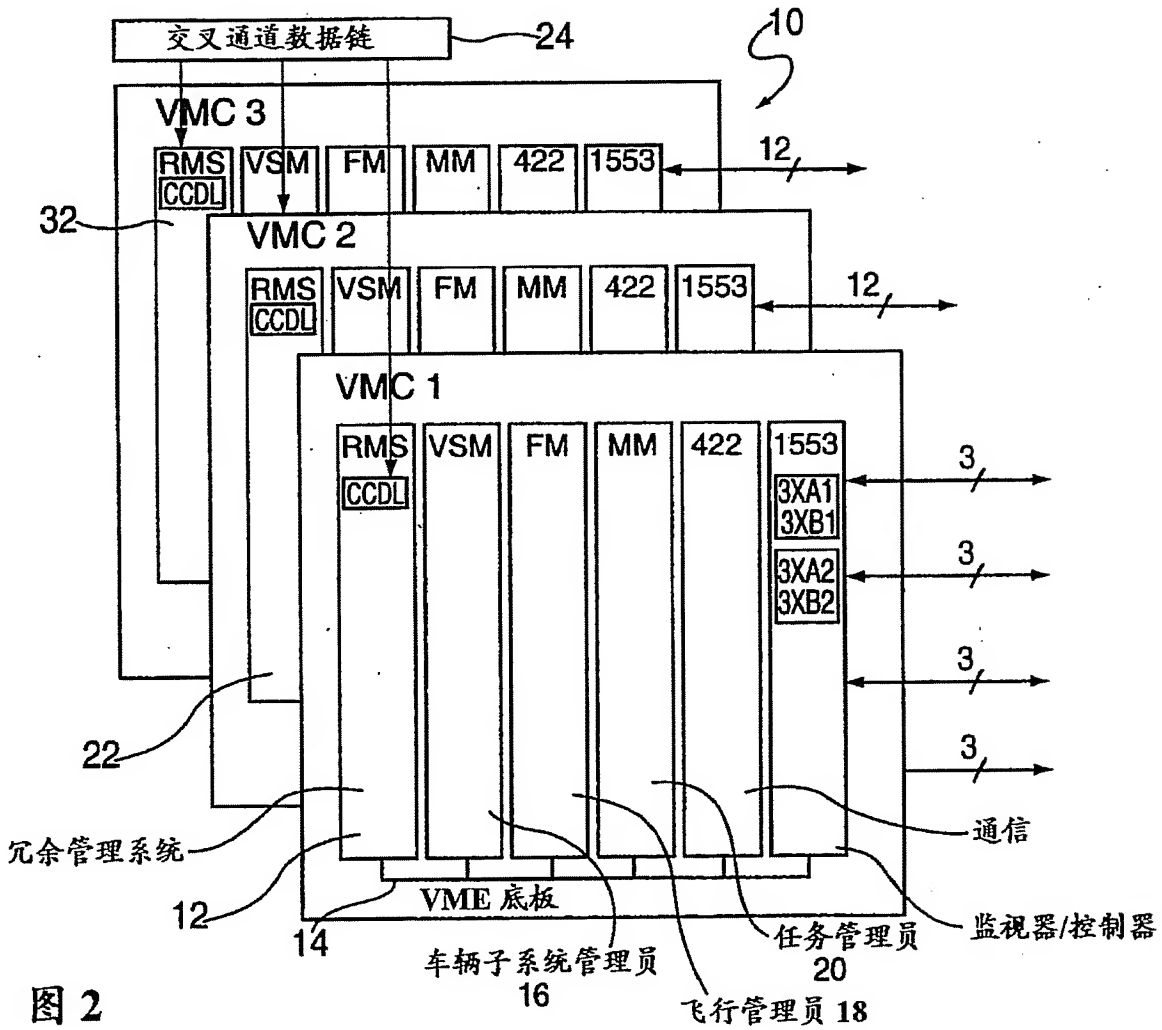

交叉通道数据链
24
10
VMC 3
RMS
CCDL
VSM
FM
MM
422
1553
12
32
VMC 2
RMS
CCDL
VSM
FM
MM
422
1553
12
VMC 1
RMS
CCDL
VSM
FM
MM
422
1553
3XA1
3XB1
3XA2
3XB2
3
3
3
3
22
冗余管理系统
12
14
VME 底板
车辆子系统管理员
16
飞行管理员 18
任务管理员
20
通信
监视器/控制器

图 2

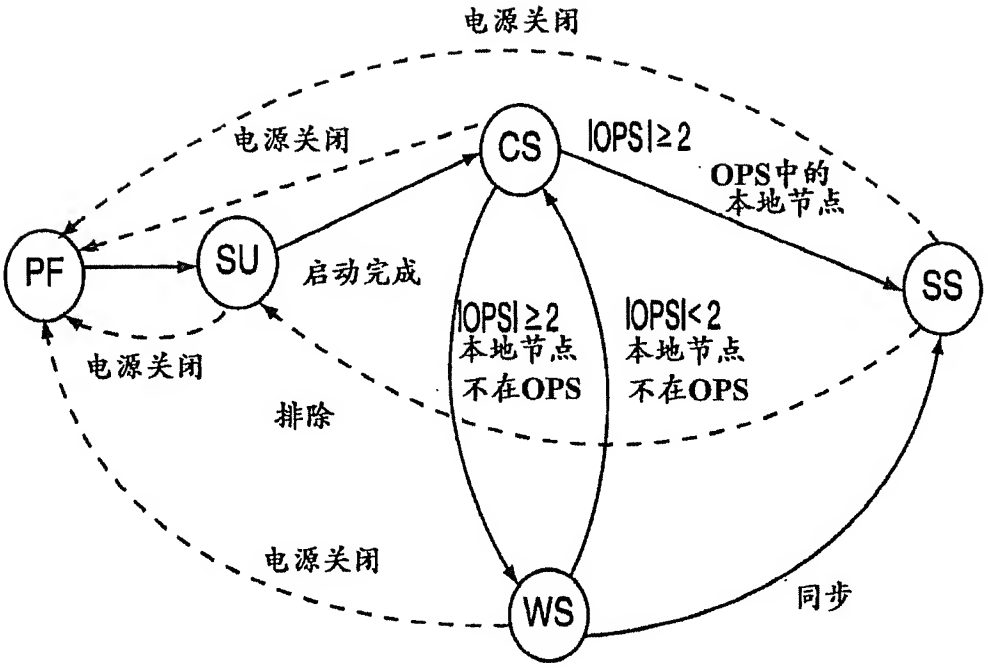

电源关闭
电源关闭
CS
|OPS|≥2
OPS中的
本地节点
PF
SU
启动完成
SS
|OPS|≥2
本地节点
不在OPS
|OPS|<2
本地节点
不在OPS
电源关闭
排除
电源关闭
WS
同步

图 3

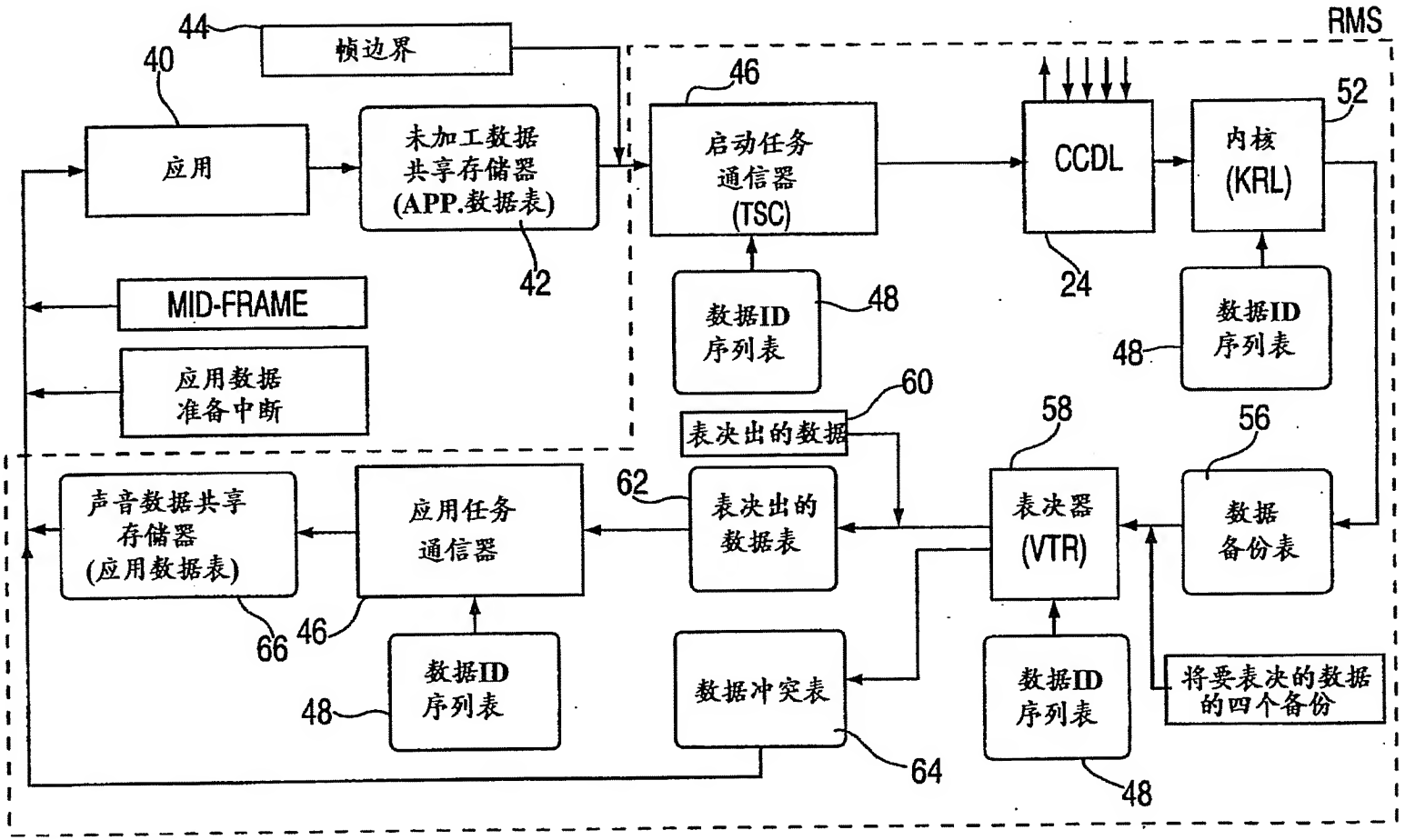

RMS
44
帧边界
40
应用
未加工数据
共享存储器
(APP.数据表)
42
MID-FRAME
应用数据
准备中断
46
启动任务
通信器
(TSC)
数据ID
序列表
48
CCDL
24
内核
(KRL)
52
数据ID
序列表
48
60
表决出的数据
58
56
62
表决出的
数据表
表决器
(VTR)
数据
备份表
声音数据共享
存储器
(应用数据表)
66
应用任务
通信器
46
数据ID
序列表
48
数据冲突表
64
数据ID
序列表
48
将要表决的数据
的四个备份

图 4

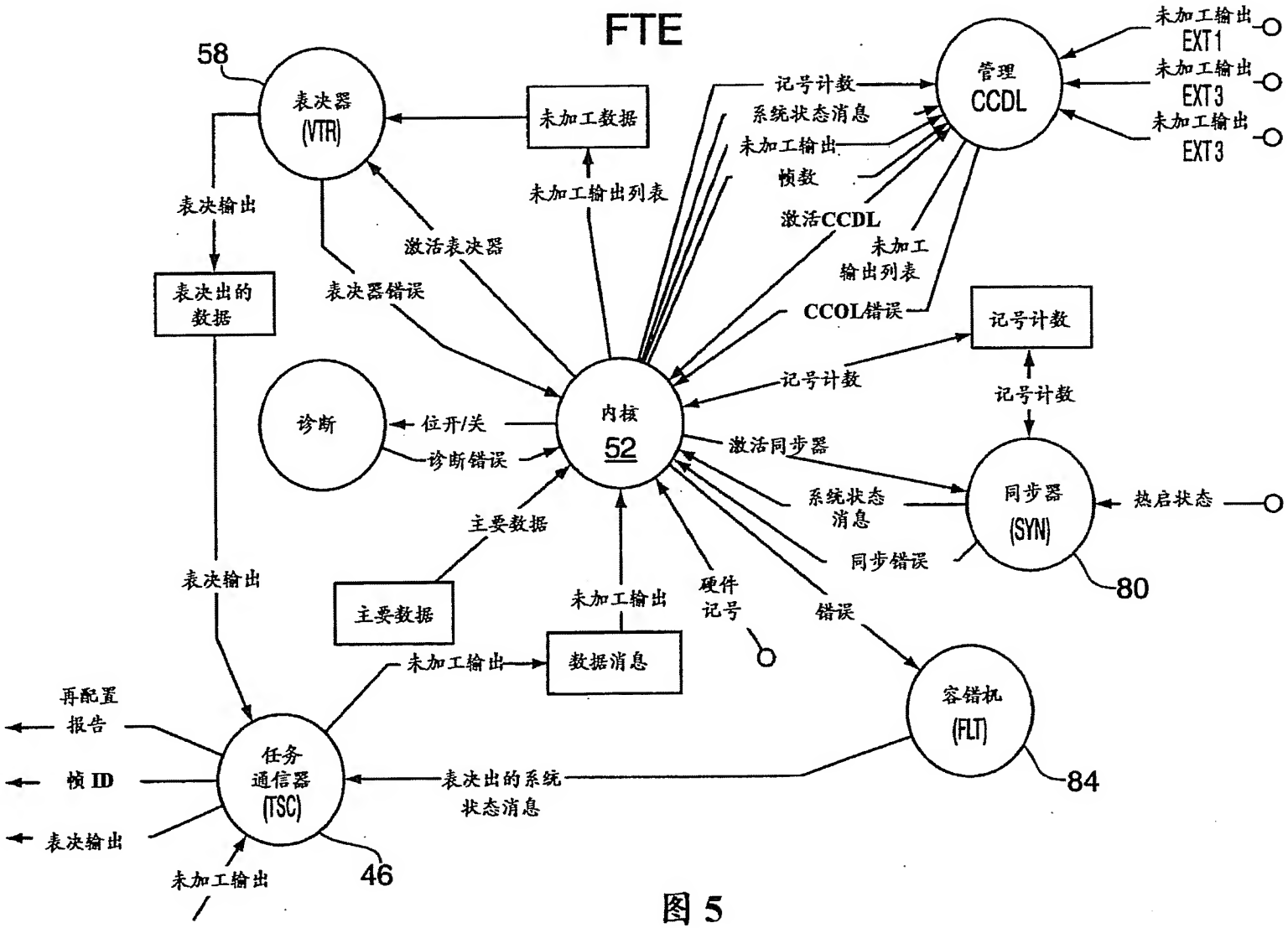

FTE
58
表决器
(VTR)
未加工数据
管理
CCDL
未加工输出
EXT1
未加工输出
EXT3
未加工输出
EXT3
记号计数
系统状态消息
未加工输出
帧数
激活CCDL
未加工
输出列表
CCOL错误
未加工输出列表
表决输出
表决出的
数据
激活表决器
表决器错误
记号计数
记号计数
记号计数
诊断
位开/关
诊断错误
内核
52
激活同步器
同步器
(SYN)
热启状态
系统状态
消息
同步错误
80
主要数据
主要数据
表决输出
未加工输出
硬件
记号
错误
数据消息
未加工输出
再配置
报告
帧 ID
表决输出
任务
通信器
(TSC)
表决出的系统
状态消息
容错机
(FLT)
84
未加工输出
46

图 5

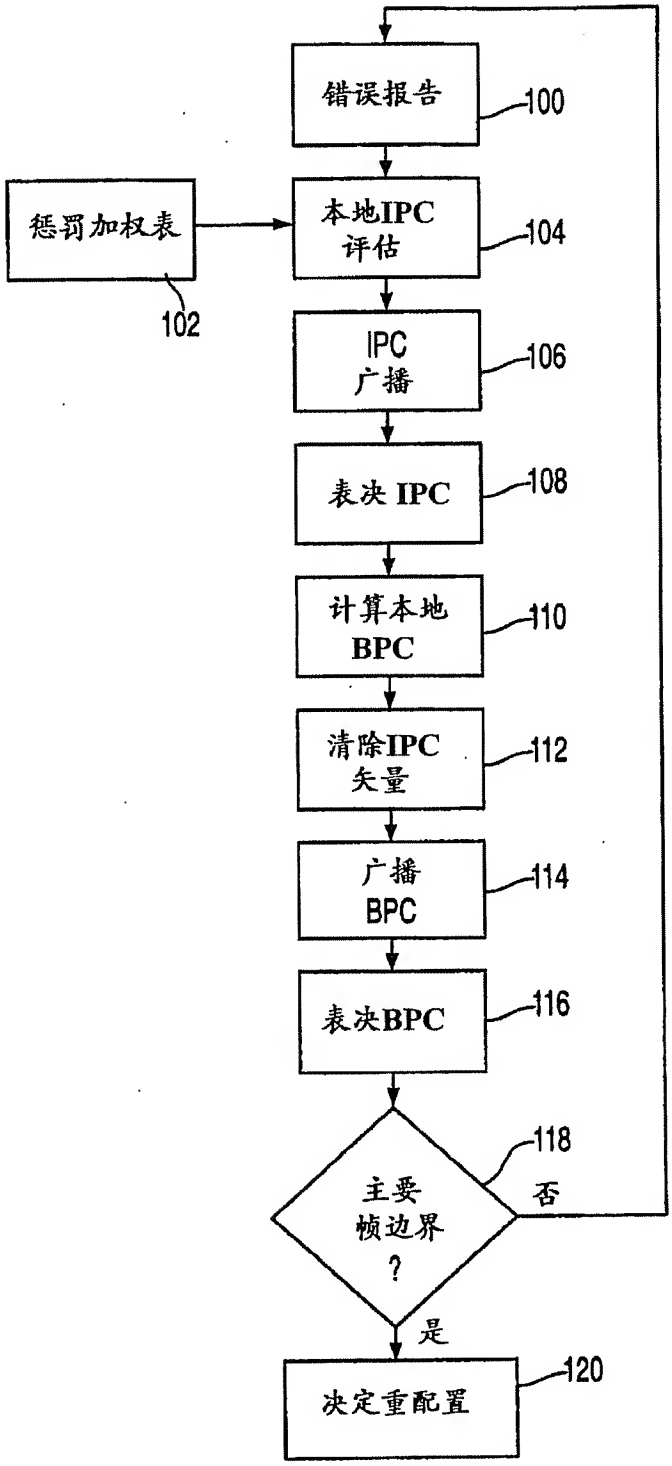

错误报告
100
惩罚加权表
本地IPC
评估
104
102
IPC
广播
106
表决 IPC
108
计算本地
BPC
110
清除IPC
矢量
112
广播
BPC
114
表决BPC
116
主要
帧边界
?
118
否
是
决定重配置
120

图 6

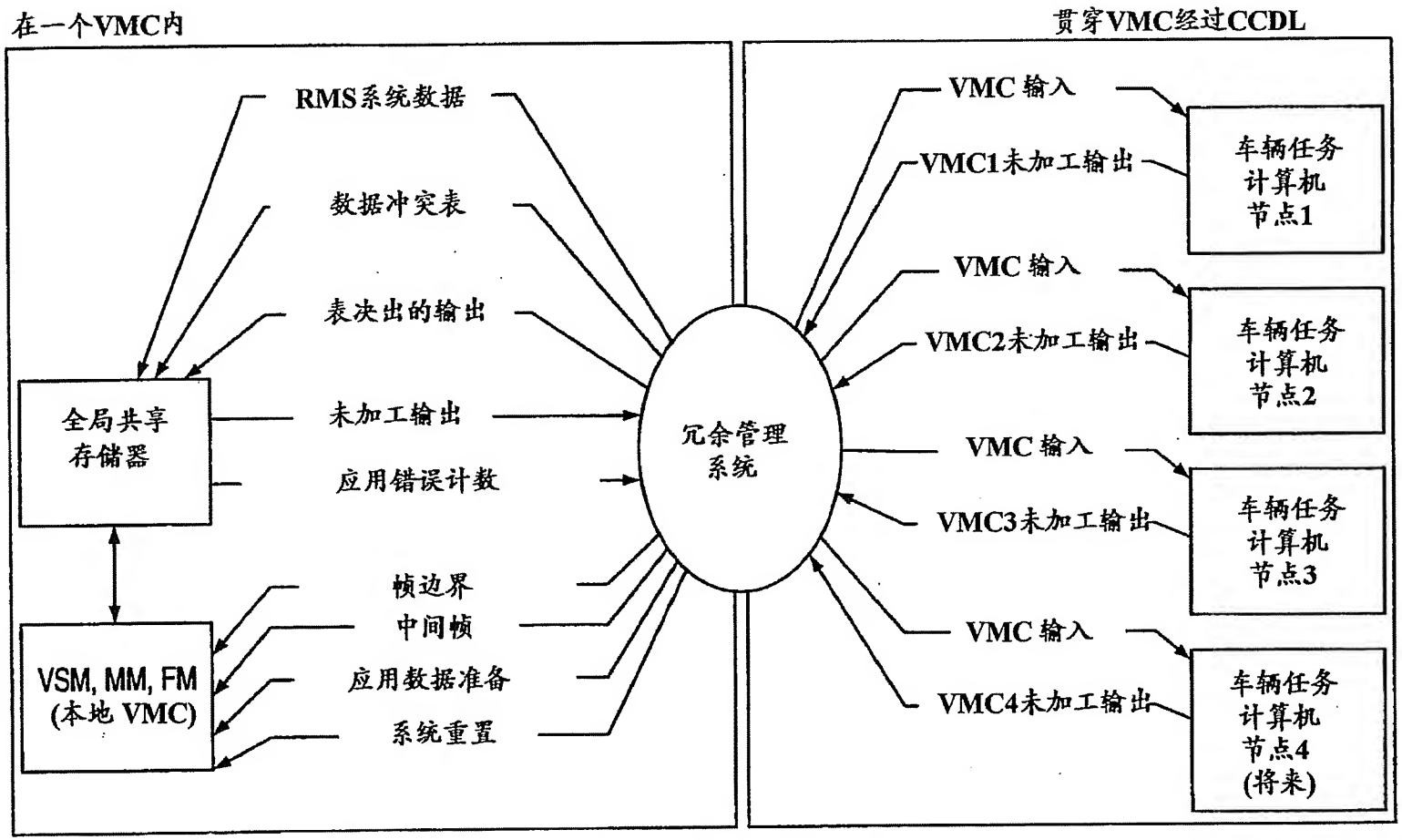

在一个VMC内
贯穿VMC经过CCDL
RMS系统数据
数据冲突表
表决出的输出
全局共享
存储器
未加工输出
应用错误计数
帧边界
中间帧
VSM, MM, FM
(本地 VMC)
应用数据准备
系统重置
冗余管理
系统
VMC 输入
VMC1未加工输出
车辆任务
计算机
节点1
VMC 输入
VMC2未加工输出
车辆任务
计算机
节点2
VMC 输入
VMC3未加工输出
车辆任务
计算机
节点3
VMC 输入
VMC4未加工输出
车辆任务
计算机
节点4
(将来)

图 7

头部

MT	NID	消息计数

消息类型0: 数据消息

MT	NID	消息计数	保留 . (16)
数据 ID			
数据值			
数据 ID			
数据值			
⋮			
数据 ID			
数据值			

消息类型1: 系统状态消息

MT	NID	消息计数	FN. BITS	保留 . (12)
ISW		NSS	CSS	RES. (8)
区间计数器				

消息类型2: 冷启动消息

MT	NID	消息计数	FN. BITS	RES. (4)	ISW
ISW0		ISW1	ISW2		ISW3
ISW4		ISW5	ISW6		ISW7

消息类型3: 错误报告和惩罚计数消息

MT	NID	消息计数	保留 . (16)	
错误矢量 0			IPC 0	BPC 0
错误矢量 1			IPC 1	BPC 1
错误矢量 2			IPC 2	BPC 2
错误矢量 3			IPC 3	BPC 3
错误矢量 4			IPC 4	BPC 4
错误矢量 5			IPC 5	BPC 5
错误矢量 6			IPC 6	BPC 6
错误矢量 7			IPC 7	BPC 7
AP/BIT 0	AP/BIT 1		AP/BIT 2	AP/BIT 3
AP/BIT 4	AP/BIT 5		AP/BIT 6	AP/BIT 7

图 8

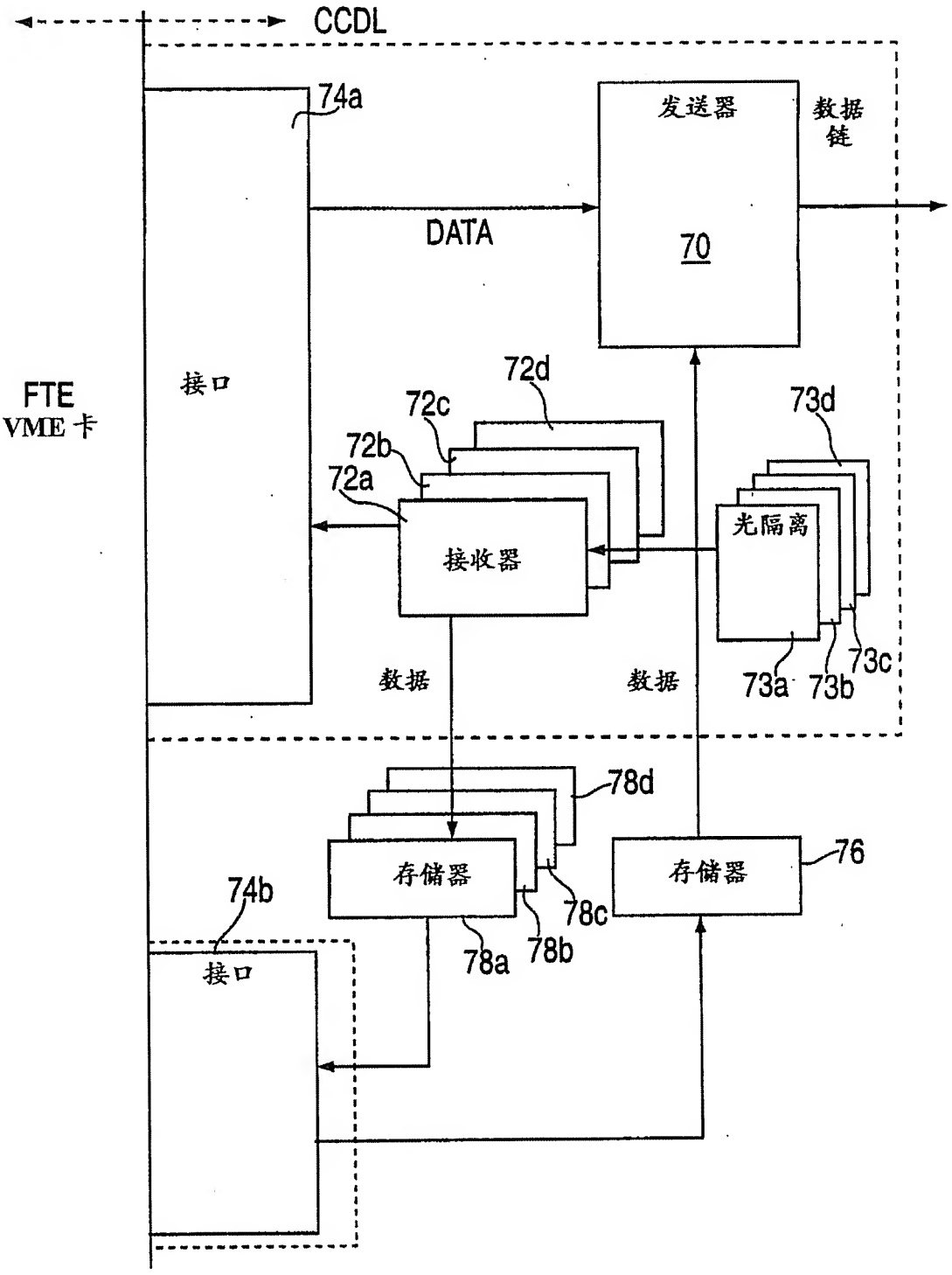

CCDL
74a
发送器
数据
链
DATA
70
FTE
VME 卡
接口
72d
72c
72b
72a
73d
光隔离
接收器
73a
73b
73c
数据
数据
78d
存储器
存储器
76
74b
78c
78a
78b
接口

图 9

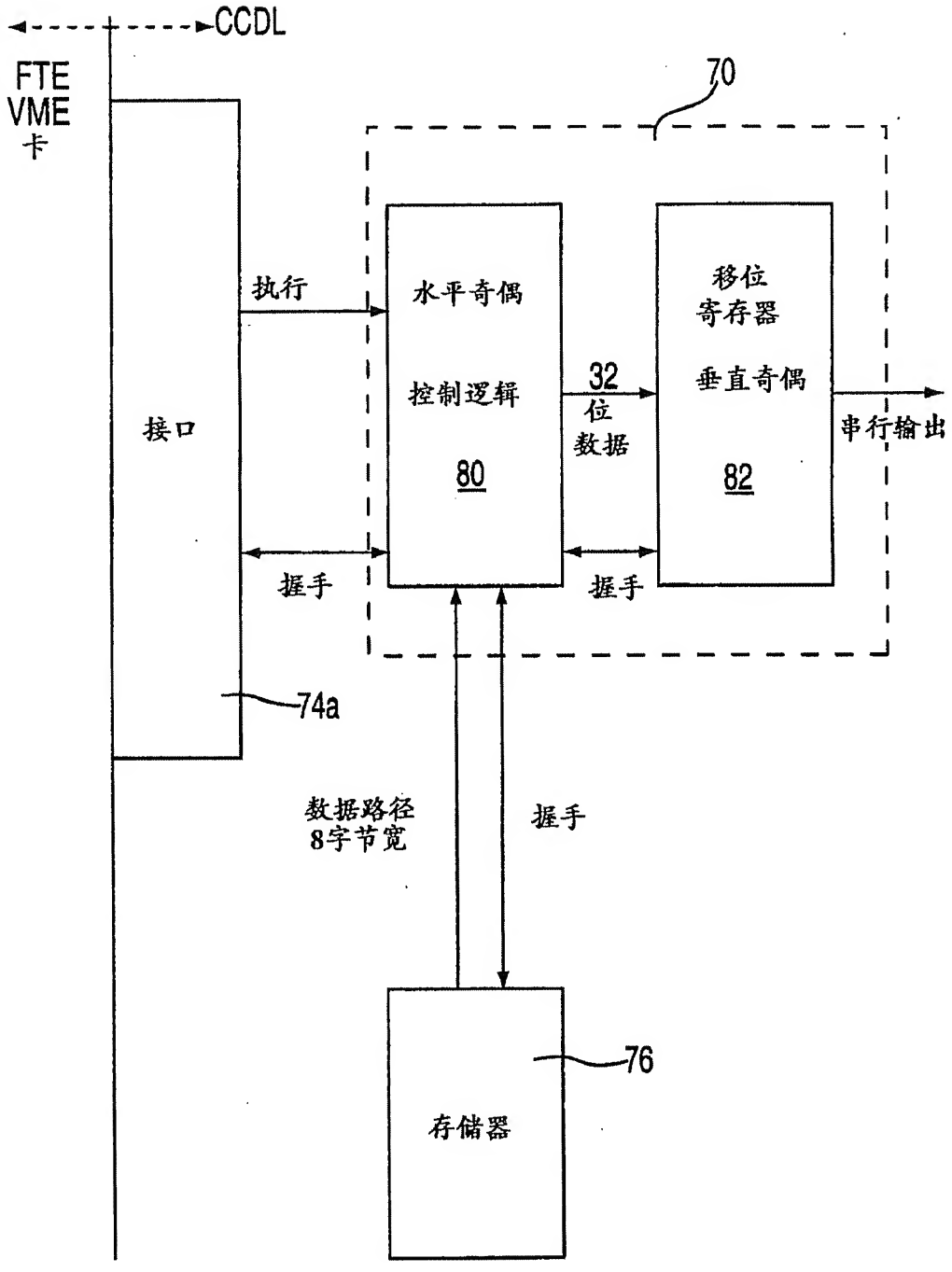

CCDL
FTE
VME
卡
70
执行
水平奇偶
控制逻辑
80
32
位
数据
移位
寄存器
垂直奇偶
82
串行输出
接口
握手
握手
74a
数据路径
8字节宽
握手
76
存储器

图 10

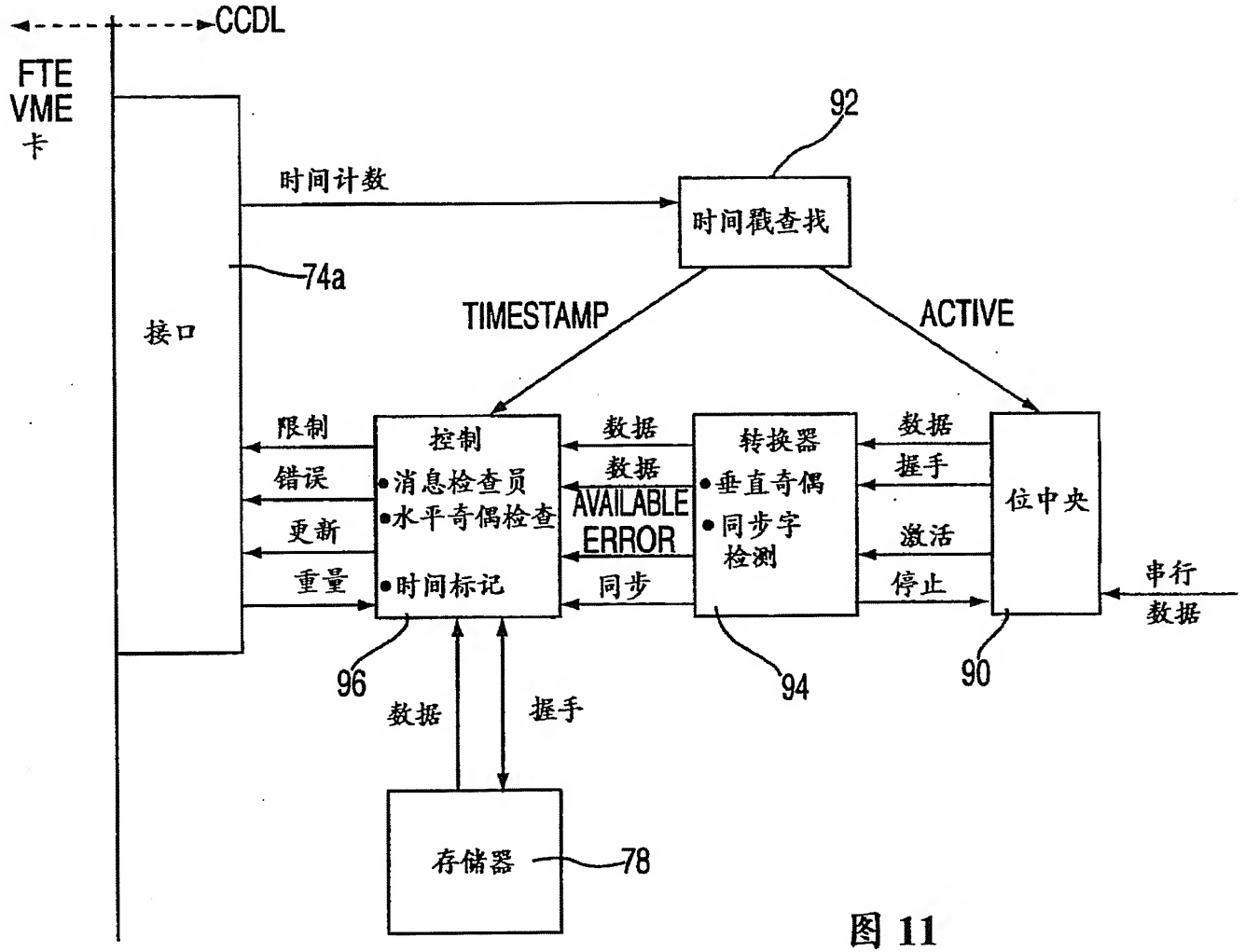

CCDL
FTE
VME
卡
92
时间计数
时间戳查找
74a
接口
TIMESTAMP
ACTIVE
限制
错误
更新
重量
控制
•消息检查员
•水平奇偶检查
•时间标记
数据
数据
AVAILABLE
ERROR
同步
转换器
•垂直奇偶
•同步字
检测
数据
握手
激活
停止
位中央
串行
数据
96
94
90
数据
握手
存储器
78

图 11